Bài 2

Logic lập trình
Các thành phần của chương trình máy tính

Nội dung

- Khai báo và sử dụng các biến và hằng
- Sử dụng các phép toán số học
- Mô-đun chương trình
- Đặc điểm của chương trình tốt

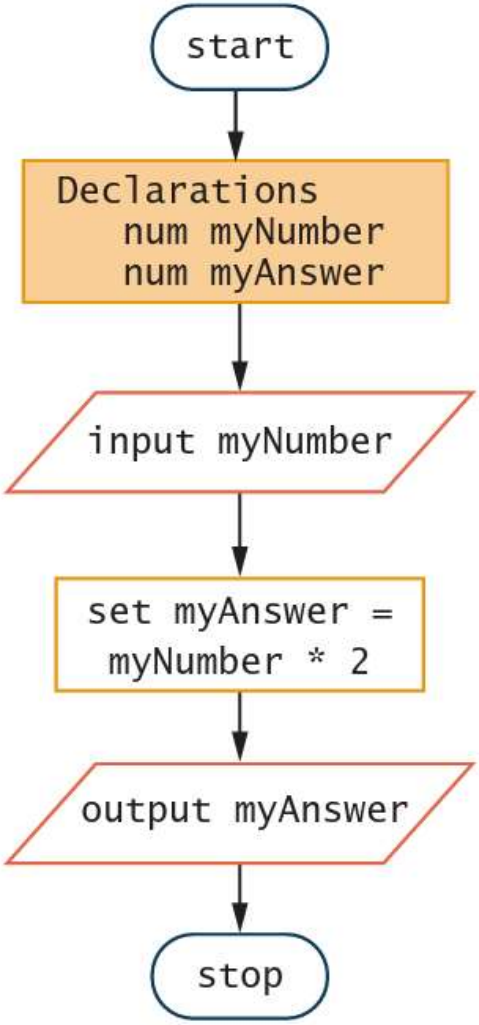
Khai báo và sử dụng các biến và hằng

- Kiểu dữ liệu cơ bản
  - Numeric (số) mô tả dữ liệu bao gồm các số.
  - String (chuỗi kí tự) mô tả dữ liệu không phải là số dùng tính toán.
- Các dạng thức của dữ liệu
  - Số nguyên (interger) và số thực / số dấu phẩy động (float).
  - Giá trị hằng số và hằng chuỗi.
    - Ví dụ, 3.14 và “Khoa Toán – Tin học”. Cách thể hiện này còn được gọi là hằng chưa đặt tên (unnamed constants).

Biến – Khai báo và khởi tạo (1)

- Biến là vị trí bộ nhớ được đặt tên
  - Dữ liệu lưu giữ tại đó có thể thay đổi.
  - Ở một thời điểm chỉ lưu giữ một giá trị dữ liệu.
  - Phải được khai báo trước khi sử dụng lần đầu tiên.
- Sự khai báo biến (declaration) là một câu lệnh cung cấp kiểu dữ liệu và tên/định danh (identifier) cho một biến.
  - Kiểu dữ liệu của biến cho biết
    - Giá trị kiểu gì có thể được lưu giữ trong biến.
    - Biến được sắp đặt trong bộ nhớ như thế nào.
    - Các phép toán / thao tác gì có thể được thực hiện với biến.

Biến – Khai báo và khởi tạo (2)




```
start
  Declarations
      num myNumber
      num myAnswer
  input myNumber
  set myAnswer = myNumber * 2
  output myAnswer
stop
```

Biến – Khai báo và khởi tạo (3)

- Sự khởi tạo biến (initialization)
  - Là khai báo biến với một giá trị bắt đầu.

```
num mySalary
num yourSalary = 14.55
string myName
string yourName = "Juanita"
```

- Giá trị rác (garbage) là giá trị chưa xác định của biến trước khi nó được khởi tạo hoặc gán giá trị cụ thể.

Biến – Đặt tên (1)

- Tên của biến cần hợp lý và mang tính mô tả.
- Mỗi ngôn ngữ lập trình có các quy tắc để đặt tên / định danh
  - Hầu hết cho phép các chữ cái và chữ số.
  - Một số cho phép dấu gạch nối _, gạch ngang –.
  - Các từ khóa dành riêng không được dùng để đặt tên vì chúng là một phần của cú pháp.
  - Giúp chương trình dễ đọc và dễ hiểu.
- Hầu hết các ngôn ngữ lập trình hiện đại, tên biến có phân biệt chữ hoa, chữ thường.

Biến – Đặt tên (2)

- Các kiểu quy ước đặt tên / định danh phổ biến
  - Kiểu lạc đà (camel case): Java, C#
  - Kiểu Pascal (Pascal case): Pascal, Visual Basic
  - Kiểu con rắn (snake case): C, C++, Python
- Tên biến trong logic chương trình nên theo 3 quy tắc sau:
  - Là một từ, chỉ chứa các chữ cái, chữ số, dấu _, dấu –.
  - Bắt đầu bằng một chữ cái.
  - Phải có ý nghĩa phù hợp.
- Kiểu lạc đà là quy ước trong đó tên biến bắt đầu bằng chữ cái thường và mọi từ tiếp theo bắt đầu bằng chữ cái hoa.
  - Ví dụ: myLastName, interestRate

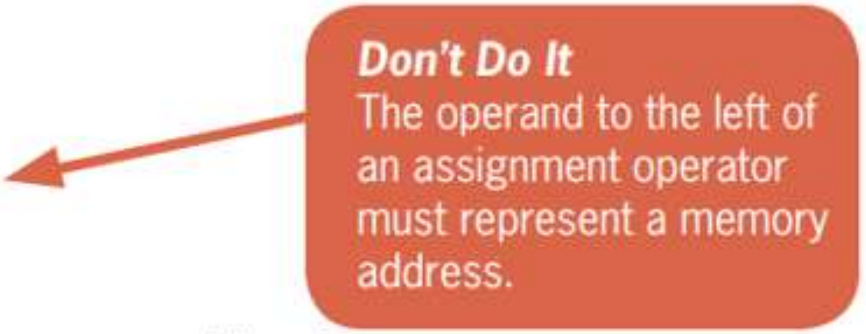
Biến – Gán giá trị (1)

- Lệnh gán (assignment statement)
  - Thực hiện việc gán một giá trị cho biến.

```
set someNumber = 2
set someNumber = 3 + 7
set someOtherNumber = someNumber
set someOtherNumber = someNumber * 5
```

```
someNumber = 2
someOtherNumber = someNumber
```

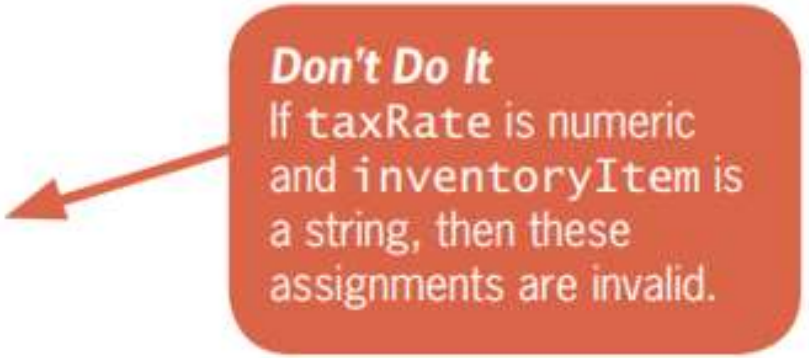
```
set 2 + 4 = someNumber
set someOtherNumber * 10 = someNumber
set someNumber + someOtherNumber = 10
```



Biến – Gán giá trị (2)

- Phép gán (assignment operator)
  - Kí hiệu là dấu =. Phép toán 2 ngôi và thực hiện từ phải sang trái.
  - Toán hạng bên phải là một giá trị hằng hoặc một biểu thức số học.
  - Toán hạng bên trái là một tên biến.
  - Giá trị bên phải và biến bên trái phải cùng kiểu dữ liệu.

```
Declarations
   num taxRate
   string inventoryItem

taxRate = “2.5”
inventoryItem = 2.5
```



Hằng được đặt tên

- Tương tự một biến nhưng giá trị của nó chỉ được gán một lần khi khai báo và không bao giờ thay đổi trong quá trình thực thi chương trình.
- Tên hằng trong logic chương trình nên theo quy tắc
  - Tất cả các chữ cái viết hoa và dấu _ để phân tách các từ.

```
Declarations
   num SALE_TAX_RATE = 0.06
```

- Hằng được đặt tên giúp chương trình dễ hiểu và dễ bảo trì.

```
taxAmount = price * 0.06
taxAmount = price * SALES_TAX_RATE
```

Sử dụng các phép toán số học (1)

- Các ngôn ngữ lập trình sử dụng các phép toán cơ bản:
  - Phép cộng (dấu +), phép trừ (dấu -), phép nhân (dấu *), phép chia (dấu /).

```
totalScore = test1 + test2
totalScore = totalScore + 10
```

- Một số ngôn ngữ hỗ trợ thêm các phép toán:
  - Phép chia lấy phần dư (dấu %), phép lũy thừa (dấu ^), …

Sử dụng các phép toán số học (2)

- Độ ưu tiên của các phép toán qui định thứ tự thực hiện các phép toán trong cùng một câu lệnh.
  - Biểu thức trong cặp dấu () trong cùng được thực hiện đầu tiên.
  - Phép * và / được thực hiện tiếp theo, từ trái sang phải.
  - Phép + và - được thực hiện tiếp theo, từ trái sang phải.
  - Phép gán được thực hiện sau cùng.
- Độ ưu tiên và tính liên kết của 5 phép toán cơ bản

Phép toán	Độ ưu tiên	Tính liên kết
=	Thấp nhất	Từ phải sang trái
+ -	Trung bình	Từ trái sang phải
* /	Cao nhất	Từ trái sang phải

Mô-đun chương trình

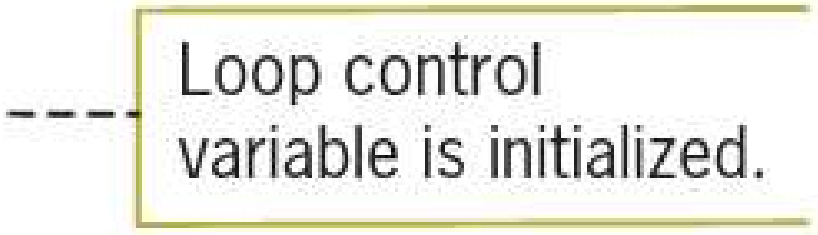
- Mô-đun là đơn vị công việc nhỏ của một bài toán lập trình.
  - Bài toán viết chương trình in bảng điểm cho một sinh viên được chia thành các đơn vị công việc:
    - Tìm danh mục các học phần đã hoàn thành.
    - Sắp xếp danh mục theo một thứ tự (mã học phần, học kỳ).
    - Hiển thị danh mục đã sắp xếp.
- Mô-đun còn được gọi là chương trình con (subroutine), thủ tục (procedure), hàm (function), phương thức (method).
  - Pascal, Visual Basic gọi là thủ tục.
  - C, C++ gọi là hàm.
  - Các ngôn ngữ hướng đối tượng như Java, C# gọi là phương thức

Đặc điểm của chương trình tốt

- Để tạo ra chương trình tốt cần thực hiện:
  - Đặt các chú thích ở vị trí thích hợp khi lập logic hay viết mã.
  - Lựa chọn các định danh một cách cẩn thận.
  - Viết các câu lệnh rõ ràng.
  - Viết lời nhắc và lặp lại đầu vào rõ ràng.
  - Tiếp tục duy trì thói quen lập trình tốt trong quá trình phát triển kỹ năng lập trình.

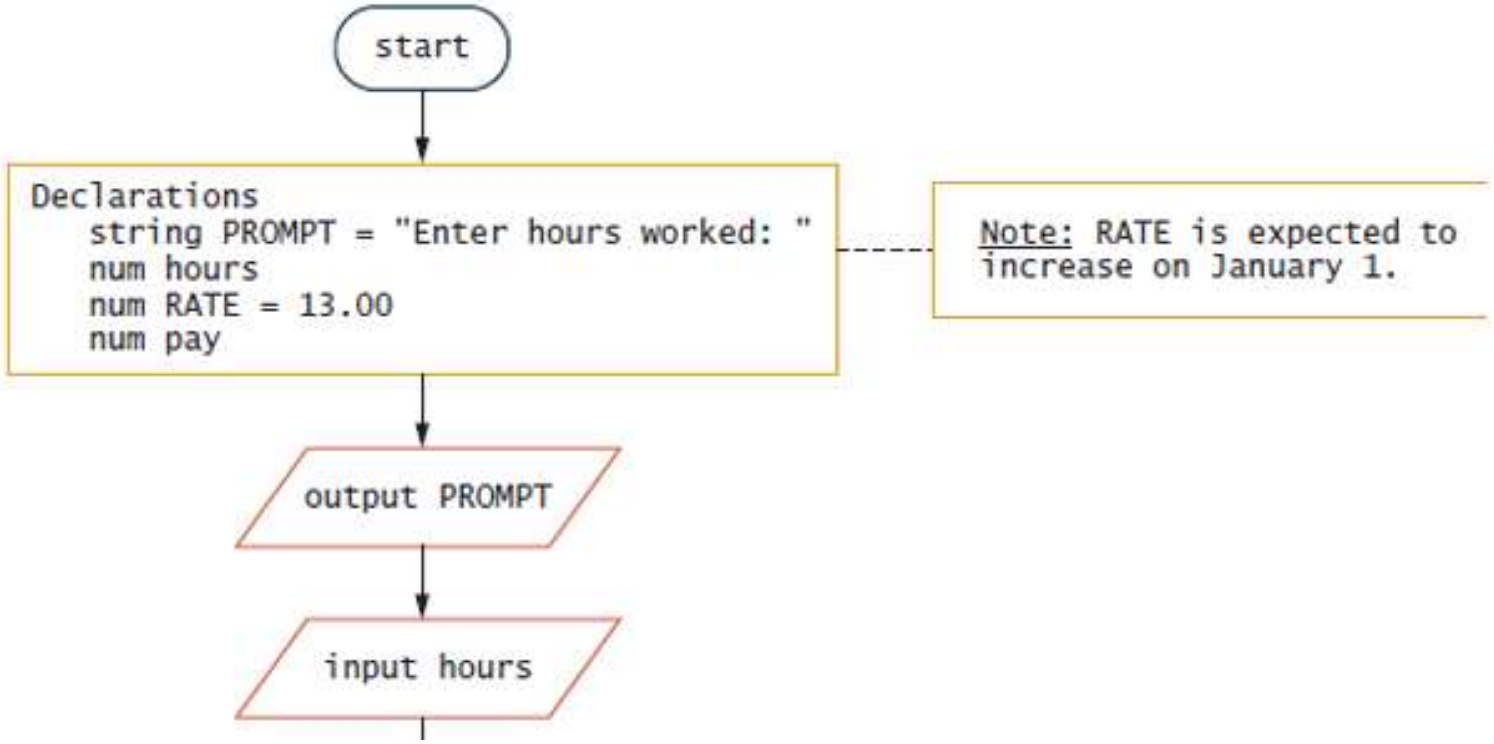
Sử dụng chú thích (1)

- Chú thích
  - Những lời giải thích bằng văn bản trong logic hay mã chương trình.
  - Không phải là một phần của logic chương trình.
  - Giúp người đọc hiểu được các câu lệnh lập trình.
- Cú pháp viết chú thích khác nhau giữa các ngôn ngữ lập trình.
- Mã giả dùng cặp dấu // để tạo chú thích.
- Lưu đồ dùng ký hiệu chú thích
  - Hình chữ nhật có 3 cạnh và đường kết nối nét đứt bên trong chứa chú thích.



Sử dụng chú thích (2)

```
Declarations
   num sqFeet
      // sqFeet is an estimate provided by the seller of the property
   num pricePerFoot
      // pricePerFoot is determined by current market conditions
   num lotPremium
      // lotPremium depends on amenities such as whether lot is waterfront
```



Lựa chọn định danh

- Các định danh tốt làm đơn giản hóa công việc lập trình và hiểu nó dễ dàng hơn.
- Một số gợi ý giúp lựa chọn định danh tốt
  - Tên biến, hằng là danh từ hoặc cụm danh từ.
  - Tên mô-đun là động từ hoặc cụm động từ.
  - Tên phải có ý nghĩa.
  - Thận trọng khi dùng với tên viết tắt.
  - Sử dụng quy ước đặt tên của ngôn ngữ lập trình để phân tách các từ cho tên dài.
  - Tên hằng sử dụng tất cả chữ cái hoa và dấu _ để phân tách các từ.

Viết các câu lệnh rõ ràng (1)

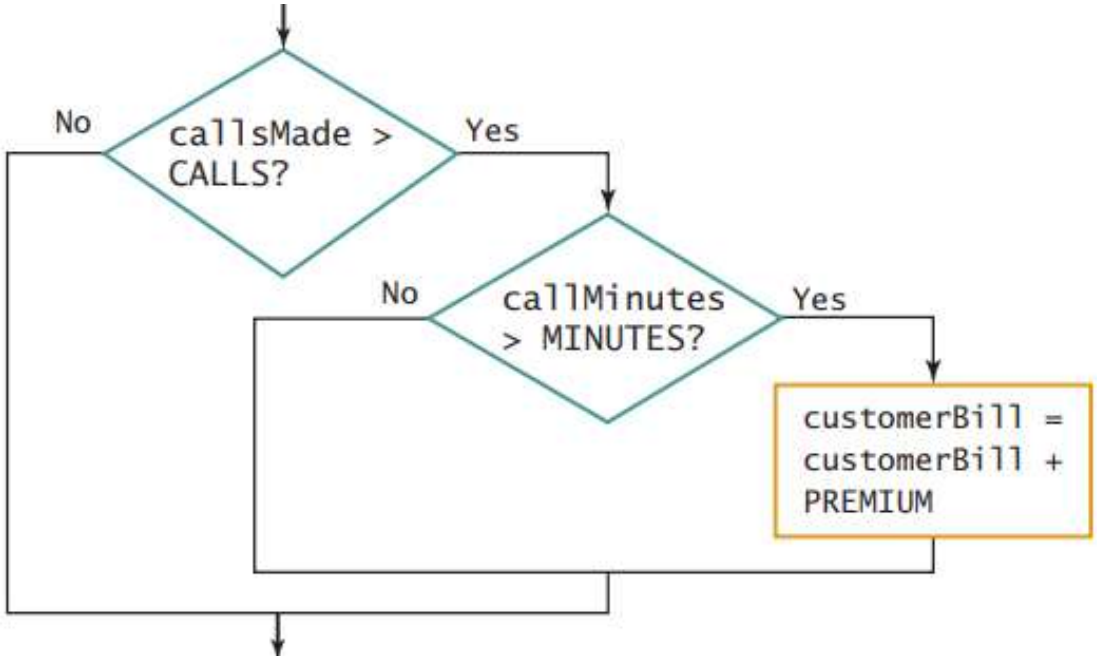
- Không nên sắp đặt nhiều câu lệnh trên cùng một dòng.
- Tránh các ngắt dòng câu lệnh dài gây khó đọc.
- Ngắt dòng sau dấu (, dấu phẩy.

```
// Line break after-punctuation
str = myString(
          reallyLongArg(),
          soManyParameters(),
          isThereSeriouslyAnotherOne()
      )
arr = {1, 2, 3, 4,
       5, 6, 7, 8}
```

Viết các câu lệnh rõ ràng (2)

- Ngắt dòng trước hoặc sau các phép toán 2 ngôi (số học, luận lý).

```
// Line break before-binary-operator
salepersonCommission = (sqFeet * pricePerFoot + lotPremium)
                       * commissionRate

// Line break after-binary-operator
salepersonCommission = (sqFeet * pricePerFoot + lotPremium) *
                       commissionRate
```



Viết các câu lệnh rõ ràng (3)

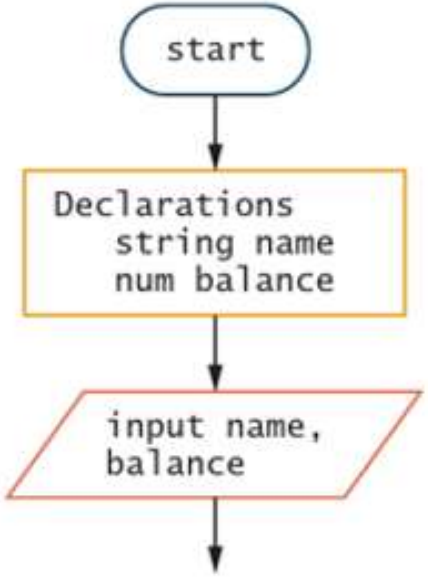
- Sử dụng biến tạm để làm rõ nghĩa câu lệnh dài

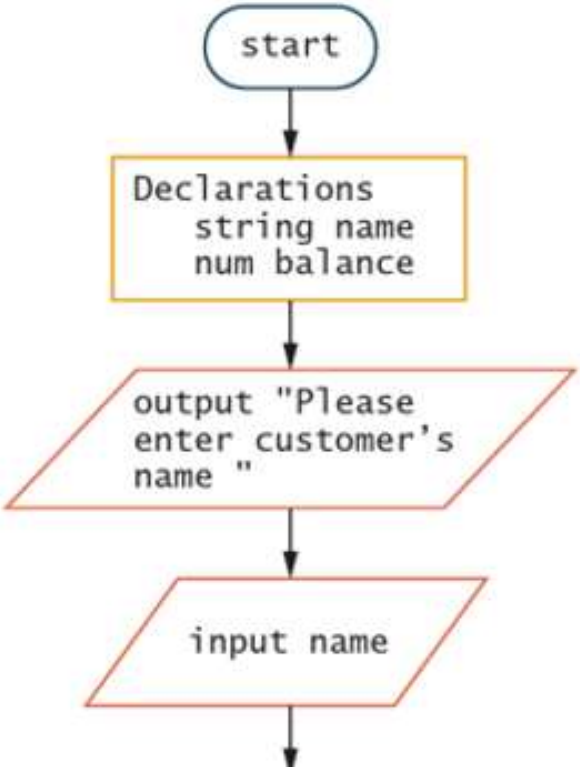
```
// Using a single statement to compute commission
salespersonCommission = (sqFeet * pricePerFoot + lotPremium) * commissionRate

// Using multiple statements to compute commission
basePropertyPrice = sqFeet * pricePerFoot
totalSalePrice = basePropertyPrice + lotPremium
salespersonCommission = totalSalePrice * commissionRate
```

- Các biến tạm hầu như không làm tốn thêm bộ nhớ và tăng thời gian thực thi chương trình.

Viết lời nhắc và nhắc lại đầu vào (1)

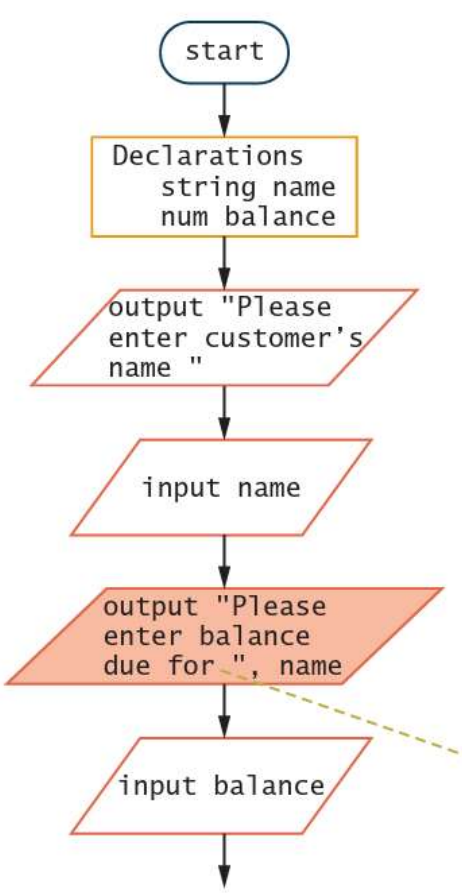
- Lời nhắc (prompt)
  - Là một thông báo được hiển thị trên màn hình để yêu cầu người dùng phản hồi.
  - Được sử dụng trong cả chương trình tương tác dòng lệnh và chương trình tương tác GUI.





Viết lời nhắc và nhắc lại đầu vào (2)

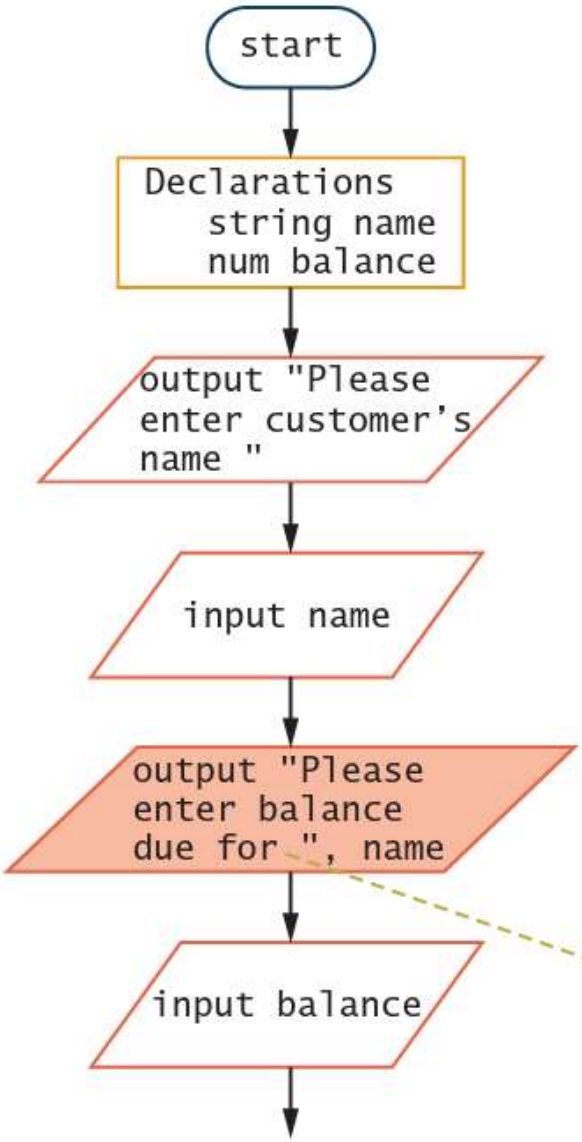
- Nhắc lại đầu vào (echoing input)
  - Là hành động thông báo lại đầu vào của người dùng trong lời nhắc kế tiếp hoặc ở đầu ra.




```
start
  Declarations
     string name
     num balance
  output "Please enter customer's name "
  input name
  output "Please enter balance due for ", name
  input balance
```

Notice the space before the quotation mark in the prompt that asks the user for a balance due. The space will appear between "for" and the last name.

Viết lời nhắc và nhắc lại đầu vào (3)




```
start
  Declarations
     string name
     num balance
  output "Please enter customer's name "
  input name
  output "Please enter balance due for ", name
  input balance
```

Notice the space before the quotation mark in the prompt that asks the user for a balance due. The space will appear between "for" and the last name.

Viết lời nhắc và nhắc lại đầu vào (4)

```
start
   Declarations
      num x
      num y
      num sum
   input x, y                          // When the user enters
   sum = x + y                         // 200 and 300.
   output x, " + ", y, " = ", sum      // The output will display
stop                                   // 200 + 300 = 500
```

Duy trì thói quen lập trình tốt

- Mỗi chương trình ta viết sẽ tốt hơn nếu ta:
  - Lập kế hoạch trước khi viết mã chương trình.
  - Duy trì thói quen vẽ lưu đồ hoặc viết mã giả đầu tiên.
  - Kiểm thử thủ công logic chương trình.
  - Suy nghĩ cẩn thận về các tên biến, mô-đun sẽ sử dụng.
  - Thiết kế các câu lệnh chương trình sao cho dễ đọc và dễ sử dụng.